Jurnal Manjemen Informatika.Volume 6 Nomor 2 Tahun 2016, 14 - 22

RANCANG BANGUN SISTEM INFORMASI HOTEL MENGGUNAKAN GRAPH DATABASE

Claudia Mei Widyayanti
D3 Manajemen Informatika, Fakultas Teknik, Universitas Negeri Surabaya, meiclaudia26@gmail.com

Andi Iwan Nurhidayat
Jurusan Teknik Informatika, Fakultas Teknik, Universitas Negeri Surabaya, andyl34ks@unesa.ac.id

Abstrak

Meningkatnya jumlah tamu yang berkunjung, persediaan kamar, hingga layanan kamar pada hotel membutuhkan pengelolaan penyimpanan *database* cukup besar. Menurut suatu penelitian jurnal *Graph Database in Large Scale Healthcare System A Proposal for Efficient Data Management and Uilization* yang menjelaskan tentang pemakaian *graph database* dalam jumlah besar untuk sistem kesehatan. Pada jurnal tersebut membanding sistem kerja *query* antara *database MySql* dengan *graph database Neo4j* yang menghasilkan pegunaan *query* pada *database Neo4j* lebih cepat dibandingkan dengan pengunaan *query* pada *database MySql* dengan pengunaan data yang kurang kompleks. Dalam sistem informasi hotel ini dibuatlah penyimpanan *database* mengunakan *database Neo4j* yang merupakan bagian dari *Graph database.* Sistem informasi hotel ini memiliki data yang cukup komplek untuk penyimpanan pemerosesnya. Dari hasil penyimpanan tersebut terciptalah waktu pemeroses yang membandingkan rata-rata waktu yang dimiliki. Rata – rata waktu yang dihasilkan di sistem informasi hotel pada *database Mysql* adalah *289ms.* Sedangkan rata-rata yang dimiliki *database Neo4j* adalah 488,*33ms.* Dari hasil tersebut pengunaan *database Neo4j* lebih lambat dibandingkan pengunaan *database MySql.* Karena pemeroses data yang dimiliki oleh sistem informasi hotel lebih kompleks dibandingkan dengan jurnal penelitian sebelumnya.
Kata Kunci : Hotel, *Graph Database, Neo4j, MySql*

Abstract

The increasing number of guests who visit, room inventory, to room service at the hotel requires a large enough database storage management. According to a study in the journal Graph Database Large Scale Healthcare System A Proposal for Efficient Data Management and Uilization explaining graph database usage in large numbers for the health system. In the journal comparing the working system between MySql database query with Neo4j graph database that generates queries on the database Neo4j Pegunaan faster than with the use of queries on database MySql with the use of less complex data. In the hotel's information system database storage made using Neo4j database that is part of the Graph database. The hotel information system has enough data to storage pemerosesnya complex. From the results of these storage creates pemeroses time comparing the average time owned. Average - Average time resulting in the hotel information system Mysql database is 289ms. While the average owned database Neo4j is 488,33ms. From these results, the use of database Neo4j slower than the use MySql database. Because pemeroses data owned by the hotel information system is more complex than previous research journal.
Keywords: *Hotel, Graph database, Neo4j, MySql*

PENDAHULUAN

Hotel merupakan fasilitas yang terpenting dalam kebutuhan *travelling*. Mengingat banyaknya jumlah tamu yang berkunjung, persediaan kamar, layanan kamar maka masalah dalam pengelolaan penyimpanan database menjadi cukup besar. Data yang semakin banyak membuka peluang dalam pembuatan sistem yang lebih efisien dalam pengorganisasian dan pengoptimalan pencarian data. Data - data yang nantinya menjadi sebuah informasi harus dapat disimpan dengan baik sehingga sewaktu-waktu data tersebut dibutuhkan dapat diakses secara cepat.

Generasi terbaru dari perangkat lunak pada suatu sistem sangat tergantung pada penggunaan sistem *database*. Menurut jurnal penelitian Yubin, Park *et al*. (2014). *Graph Database in Large Scale Healthcare System A Proposal for Efficient Data Management and Uilization. University of Texas at Austin TX. USA.* Menjelaskan tentang pemakaian *graph database* dalam jumlah besar untuk sistem kesehatan. Jurnal tersebut membanding sistem kerja *query* antara *database MySql* dengan *graph database Neo4j.* Hasil dari penelitian tersebut menjelaskan pegunaan *query* pada *database Neo4j* lebih cepat dan stabil dibandingkan dengan pengunaan *query* pada *database MySql.* baik dalam volume dan keterkaitan. Salah satu model *database NoSQL* adalah *Graph Database.*

Dengan mulai meningkat data *Graph Database* adalah basis data yang menggunakan struktur grafik yang berisi *node*, *relasi*, dan *property* untuk mewakili dan menyimpan informasi. Keunggulan lain dalam *GraphDB* adalah biasanya lintasan grafik digunakan sebagai pengganti operasi *join* yang berpengaruh dalam *efisiensi query. Neo4j* merupakan jenis dari *GraphDB*. Pada sistem ini dibuatlah sistem informasi hotel menggunakan *Graph Database (Neo4j).*

KAJIAN PUSTAKA

Graph Database

Basis data grafik (*GraphDB*) adalah basis data yang menggunakan struktur grafik yang berisi *node*, relasi, dan *property* untuk mewakili dan menyimpan informasi. *GraphDB* diperlukan untuk data grafik yang berskala besar, terutama yang dipergunakan oleh para peneliti *jbiologi* jaringan dan situs jaringan sosial, seperti *Facebook*, dan *Twitter*. *GraphDB* memetakan secara langsung objek ke aplikasi dan lebih intuitif untuk menggambarkan *data set* asosiatif.

Beberapa keuntungan dari *GraphDB* adalah *Intuitive*, dimengerti oleh pikiran manusia, yaitu menggambarkan *entitas* dan hubungan sebagai grafik masalah umum yang akrab dengan manusia; *Elemental* untuk ilmu komputer, yaitu grafik, terutama grafik pohon (seperti *binary-tree*, *B+ tree*, *red-black tree*) berfungsi sebagai struktur data dasar dalam ilmu komputer dan berbagai masalah (*shortest path* dan *max-flow*) dapat diubah dan diselesaikan dengan algoritma grafik; *Ubiquitous*, yaitu pemodelan *ER* ke model jejaring sosial selalu dikelilingi oleh grafik baik di komputer ataupun dalam kenyataan.

Keunggulan lain dalam *GraphDB* adalah biasanya lintasan grafik digunakan sebagai pengganti operasi *join* yang berpengaruh dalam efisiensi *query*. *GraphDB* juga tergantung pada kurangnya *schema* yang kaku di mana suatu *schema* dapat selalu di ubah dengan mudah pada grafik, karena struktur grafik sendiri cukup fleksibel untuk mewakili perubahan melalui edit relasi dan properti. *GraphDB* juga dapat mendukung semua fitur basis data yang kuat .

Neo4j

(Abidin Ali, D. R. , 2012) *Neo4J* adalah system *graph database* yang memenuhi kriteria *ACID (atomicity, consistency, isolation, durability),* bersifat *transversal framework* yang menyediakan manajemen *graph database* dalam *nodes* dan *relationship*, *open source graph database* yang didukung secara komersial. *Tools* ini dirancang dan dibangun dari awal untuk menciptakan *database* yang dapat diandalkan dan dioptimalkan untuk grafik struktur.

Neo4j telah dikembangakan sejak tahun 2003, dapat ditulis dengan Bahasa *Java, Ruby, Scala, Python, Clojure* dan sebagainya. *Neo4j* adalah sebuah *graph database* yang kuat, *scalable*, dan *high-performance* yang memiliki fitur dapat menampung milyaran *node* dan relasi, memiliki kemampuan menjelajahi *graph* dengan cepat, memiliki bahasa *query* untuk *graph*, dan dapat mendukung transaksi seperti pada *Relational Database.*(The Neo4j Team, 2013)

Pada *Neo4J* data direpresentasikan dalam bentuk *Node* dan Relasi. *Node* merupakan titik. Relasi adalah penghubung antar dua *node*. Baik *Node* dan Relasi memiliki *property* yang bisa kita sesuaikan. Sebelum menciptakan *node* atau Relasi, bisa kita definisikan atribut yang ada di tiap *node* atau Relasi tersebut. Hal ini opsional tidak wajib tapi merupakan best practice untuk dilakukan.

Perancangan database di *Neo4J* hanya sebatas desain *node* dan relasi. Tidak ada istilah tabel atau *database* dalam *Neo4J*. Hal ini berbeda dengan basis data *RDBMS* atau *NoSQL* lainnya. Untuk *Neo4J*, dalam satu *database server Neo4J* hanya ada satu *database*, kita tidak perlu mendefinisikan nama *database*, juga tidak ada tabel didalam *database* tersebut. Sehingga ketika kita ingin mengubah *database* maka seluruh file di direktori data di *Neo4J* harus dihapus secara manual. Cara ini yang paling cepat dibanding menghapus *node* dan relasi.

Query Cypher Language

Query database dengan bahasa *query Cypher*. Ini adalah bahasa *query* yang baru-baru ini ditambahkan ke *Neo4J*. *Cypher* adalah, bahasa *SQL* yang terinspirasi untuk menggambarkan pola di grafik menggunakan s*intaks*. Hal ini memungkinkan untuk menyatakan apa yang ingin di pilih, menambahkan, memperbarui atau menghapus dari data grafik. *Cypher* terinspirasi oleh sejumlah pendekatan yang berbeda dan dibangun berdasarkan praktek didirikan untuk *query*. Sebagian besar kata kunci seperti *WHERE* dan *ORDER BY* terinspirasi oleh *SQL*. *Cypher* menerapkan struktur dari *SQL query* yang dibangun menggunakan berbagai klausa.

METODE

Analisa Sistem

Pada tahap ini merupakan tahap dilakukannya analisa terhadap sistem berjalan (sistem lama), kemudian dibuatkan sistem usulan (sistem baru) yang akan digunakan nantinya di Hotel. Sedangkan untuk sistem usulan yang akan dibuat menggunakan *graph database* pada sistem basis datanya.

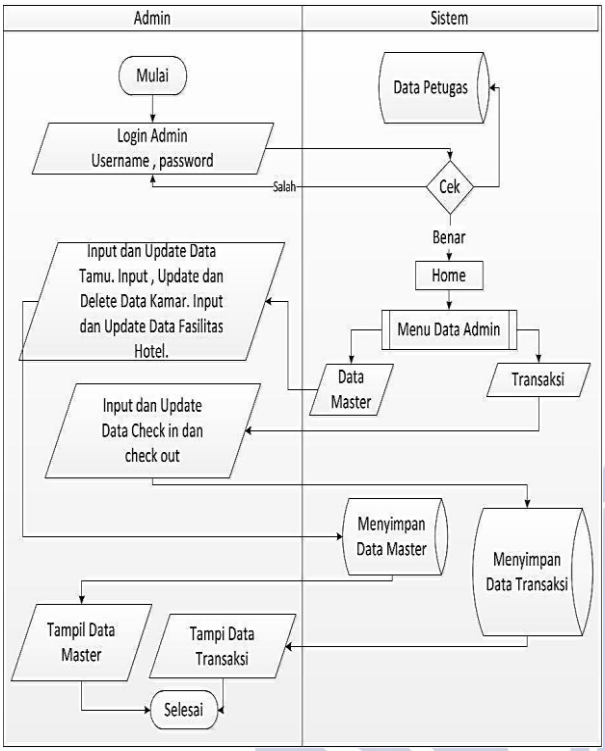
Perancangan Sistem Informasi Perhotelan ini akan dikembangkan sistem informasi berbasis web. Sehingga nantinya apabila data tersebut dibutuhkan dapat dicari melalui pencarian data dan dapat di cetak. Data yang telah tersimpan dalam sistem juga dapat di ubah dan di hapus sehingga memudahkan manajemen hotel untuk melihat laporan data yang dibutuhkan.

Berikut ini adalah gambar *flowmap* sistem baru yang diajukan penulis:

1. Flowmap pada sistem informasi hotel

a. Diagram Flowmap Proses Petugas Admin

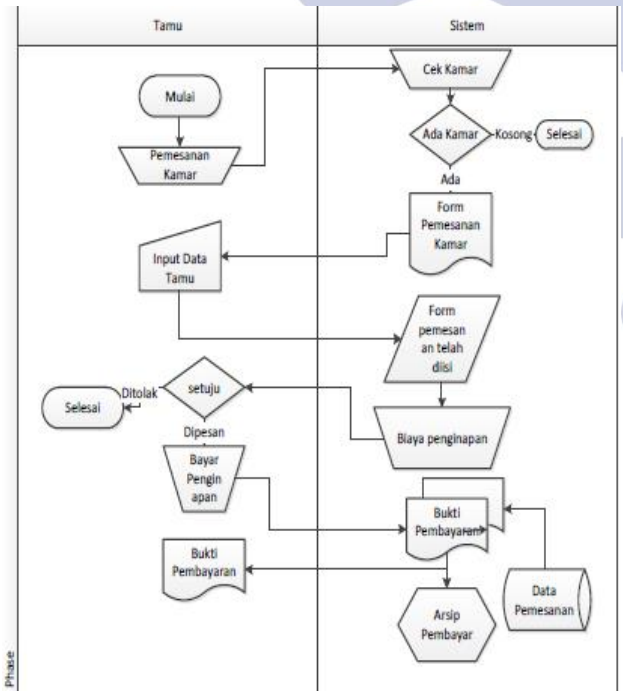
Pada flowmap petugas admin (gambar 1) menjelaskan admin harus memiliki username dan password untuk login pada menu admin hotel. Apabila admin belum memiliki username dan password maka proses gagal. Setelah admin berhasil login admin akan dapat melakukan menambahkan, menghapus dan memperbarui data master yang meliputi data tamu hotel, data kamar hotel, data tipe kamar pada hotel. Dan juga admin dapat menambahkan, menghapus dan memperbarui data transaksi yang ada pada hotel. Data transaksi meliputi data *reservasi* yang dilakukan oleh tamu apabila datang langsung kehotel, data *check – in* tamu dan data *check – out* tamu.



Gambar 1. Flowmap proses petugas admin

b. Diagram Flowmap Proses Data Reservasi Hotel

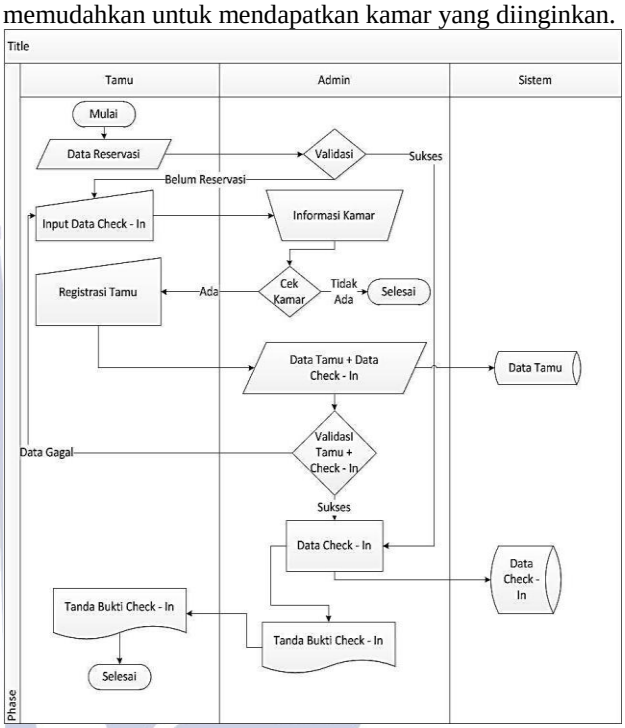
pada Flowmap *reservasi* (gambar 2). Disini Tamu dapat melakukan reservasi hotel terlebih dahulu. Proses reservasi ini memudahkan tamu untuk memesan hotel jauh hari sebelum check – in. Dan tamu dapat memilih jenis tipe kamar yang tersisa dengan mudah.



Gambar 2. Flowmap *reservasi*

c. Diagram Flowmap Proses Data Check – in

Sebelum tamu menempati kamar hotel tamu wajib *Check – In* untuk mendapatkan kode *Check – In*. Kode Check – In ini nanti akan memudahkan proses penginputan data check-out selanjutnya. Pada proses check-in juga akan membatu admin untuk menghitung jumlah tamu check-in sewaktu waktu. Apabila tamu memiliki kode reservasi maka akan memudahkan untuk mendapatkan kamar yang diinginkan.



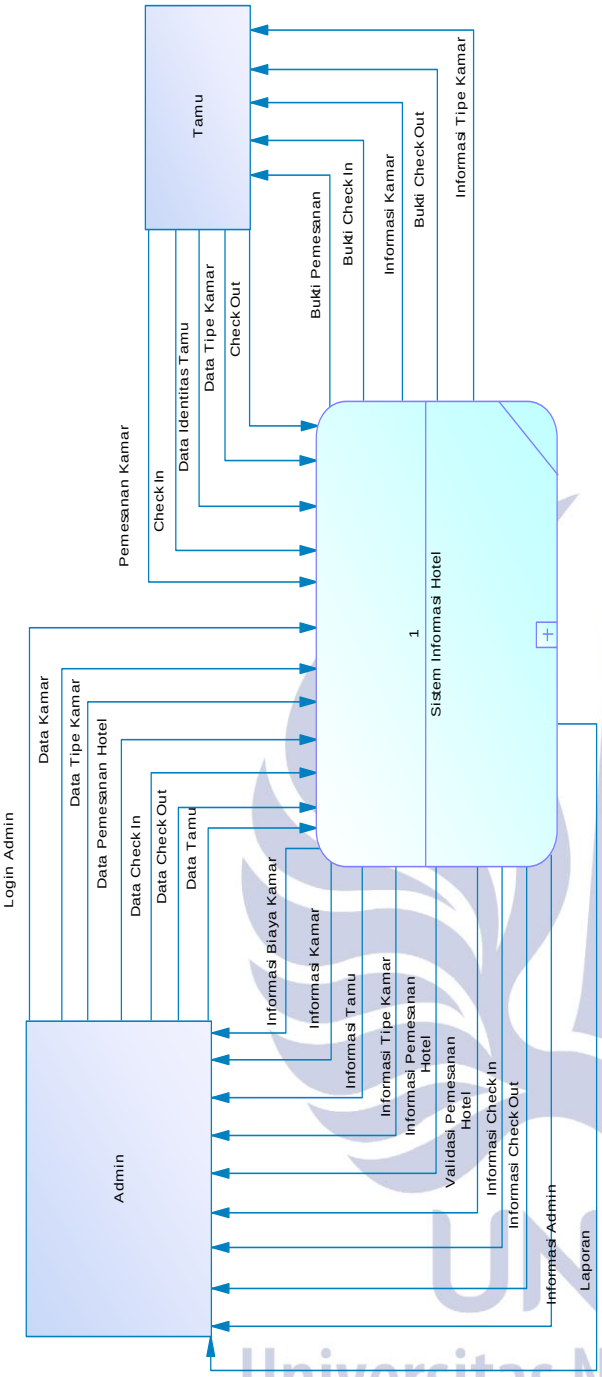
Gambar 3. Flowmap proses c*heck-in*

DESAIN SISTEM

Desain Model

1. DFD (Data Flow Diagram) Level Konteks

Pada Diagram Kontext Sistem Informasi Hotel ini terdapat dua pengguna, yaitu admin dapat menggunakan sistem ini dengan memasukkan data kamar, data tipe kamar, data *reservasi* hotel, data *check in*, data *check out*, data tamu, memvalidasi *reservasi* hotel dan menampilkan semua informasi, laporan yang ada pada hotel. Untuk hak akses yang kedua yaitu tamu, pada sistem ini tamu dapat memasukkan pemesanan kamar, identitas tamu, menerima bukti pemesanan, *check in, check out*, dan menerima informasi kamar ataupun tipe kamar. Diagram *contex* dapat dilihat pada gambar 4. Pada Diagram



Gambar 4. DFD Level Konteks

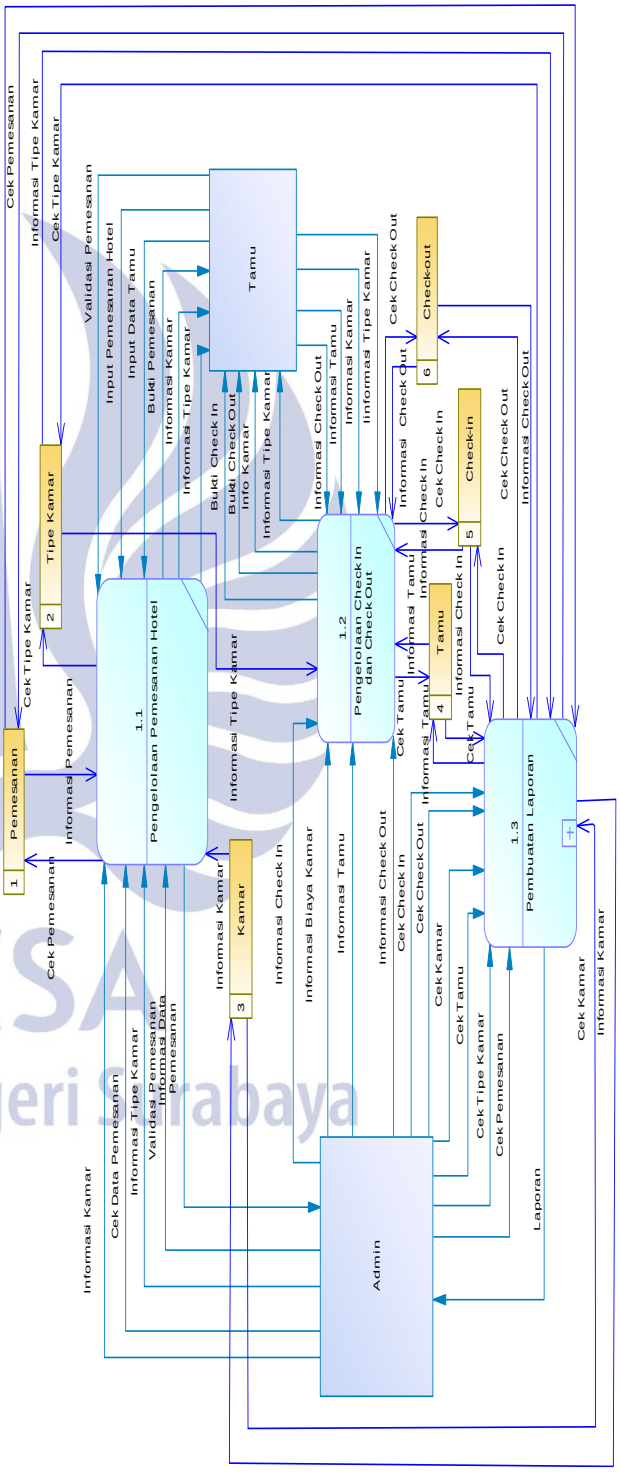
2. DFD Level 1

Pada DFD level 1 (gambar 5) terdapat beberapa proses diantaranya :

Pengelolaan pemesanan hotel, merupakan proses melakukan pemesanan hotel yaitu, admin dapat melihat informasi kamar, tipe kamar, data pemesanan, dapat mengecek data pemesanan, dan memvalidasi pemesanan. Sedangkan tamu dapat melakukan pemesanan hotel, memasukkan data tamu, menerima bukti pemesanan, memvalidasi pemesanan, dan mendapatkan informasi kamar dan tipe kamar.

Pengelolaan *check in* dan *check out*, pada proses ini admin dapat mengelola informasi *check in*, informasi biaya kamar, informasi tamu, informasi *check out*, mengecek kamar, mengecek tamu, mengecek tipe kamar, dan mengecek pemesanan. Sedangkan untuk hak akses tamu, tamu memperoleh informasi kamar, tipe kamar, bukti *check in*, bukti *check out*, memasukkan proses *check in*, identitas tamu, memilih kamar, dan memilih tipe kamar.

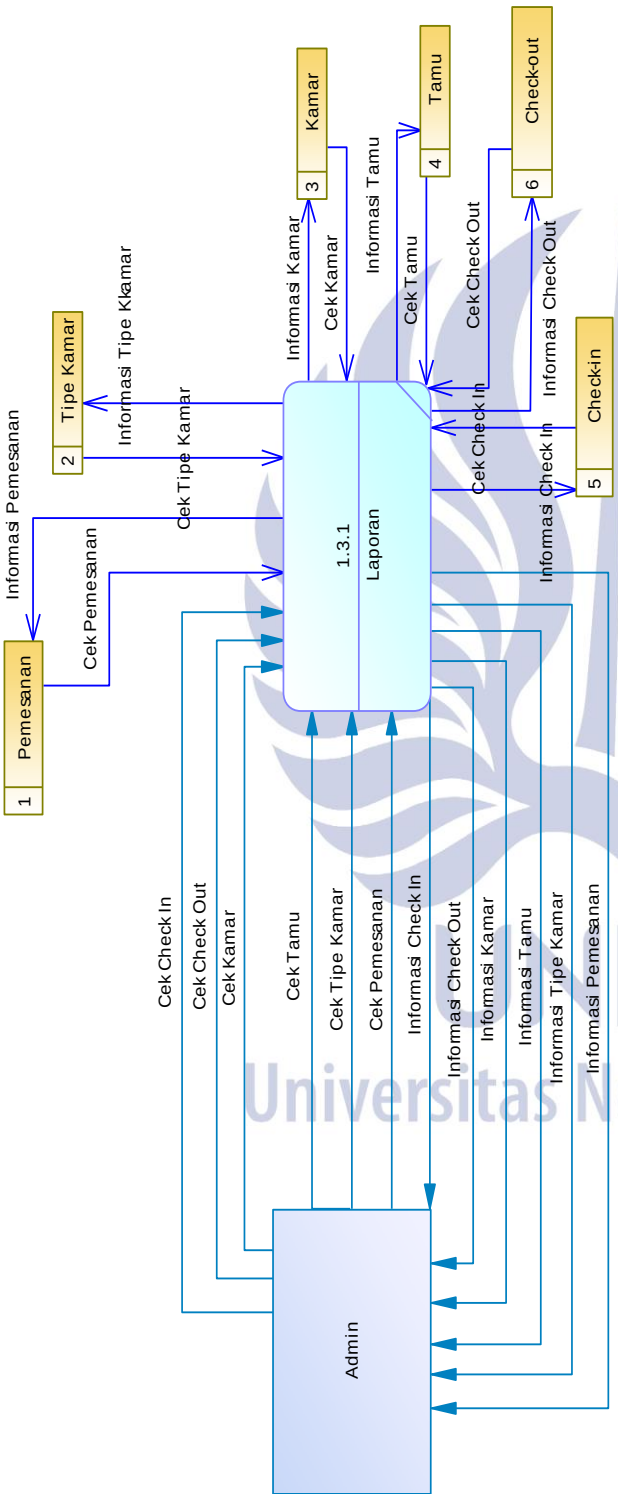
Pembuatan laporan, pada proses ini admin dapat melihat laporan dari semua proses yang dilakukan di sistem ini.



Gambar 5. DFD Level 1

3. **DFD level 2**

Pada DFD level 2 (Gambar 6) menjelaskan tentang proses pembuatan laporan yang dilakukan oleh admin. Pada level ini dapat menghasilkan semua laporan dari data data yang telah ada. Admin dapat mengecek semua data dari hasil informasi yang ada pada laporan. Laporan yang dihasilkan pada sistem informasi hotel tersebut, Laporan *check-in*, *check-out*, *reservasi*, tipe kamar, kamar, dan tamu.



Gambar 6. DFD Level 2

4. Desain *Database*

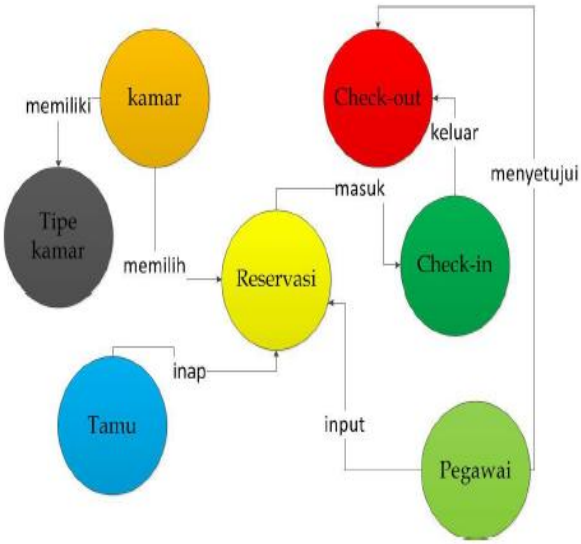
Pada gambar 7 menjelaskan tentang desain *graph database* untuk sistem informasi hotel. Pada gambar gambar 7 terdapat *node* yang saling berelasi. Setiap *node* terhubung ke *node* lain melalui garis yang memiliki arah yang disebut sebagai relasi. Baik *node* maupun relasi boleh memiliki *properties*. Berkebalikan dari database relasional, nilai *properties* pada *node* atau relasi akan lebih baik bila sebisa mungkin di-*'normalisasi'* menjadi *node* sehingga bisa dipakai secara efisien di query. Berikut adalah penjelasan *node* pada gambar 7.

Pada *node* Kamar berelasi dengan *node* TipeKamar. *Node* kamar dengan *node* TipeKamar dihubungkan dengan relasi memiliki. Dalam *node* Kamar apabila dipangil propertis yang ada pada *node* Kamar maka otomatis juga akan menampilkan salah satu propertis yang ada pada *node* TipeKamar.

Pada *node* Kamar dengan *node reservasi* dihubungkan dengan relasi memilih. Dalam *node reservasi* akan dapat memangil salah satu properties yang ada dalam *node* kamar.
Pada *node* tamu berelasi dengan *node reservasi* yang dihubungan dengan relasi inap. Dalam *node reservasi* otomatis akan dapat memangil properties yang ada dalam *node* tamu.

Pada *node* pegawai berelasi dengan *node reservasi* yang dihungkan dengan relasi input. Dalam *node reservasi* juga akan dapat memanggil propertis ada dalam *node* pegawai.

Pada *node reservasi* juga berhubungkan dengan *node check in* yang dihubungkan dengan relasi masuk. Pada propertis *node reservasi* yang nanti akan dipangil dalam *node ckeck in.* Pada *node check in* memiliki hubungan relasi keluar dengan *node check out. Pada* node *pegawai juga berelasi dengan* node check out.



Gambar 7. Desain *Database* Hotel

5. Perbandingan antara query *database relasional MySql* dengan *database neo4j.*

Berikut adalah beberapa *query* pada *database Mysql* dengan *Neo4j:*

a. Proses menambah data

1) Pada *MySql* :

```
INSERT INTO `hotel`.`kamar` (`ID_KAMAR`, `ID_TIPE`, `NOMOR_KAMAR`, `NAMA_KAMAR`, `STATUS`, `KAPASITASDEWAS`, `KET`, `HARGA`, `PCT`) VALUES ('2', '1', 'K2', 'standart room', 'Free', '2 orang', 'abcd', '300000', 'a.jpg');
```

2) Pada *Neo4j:*

```
create (n:kamar{nomor:'K1',namakmr:'Single Room', status:'free', kapasitas:'1 orang', harga:'230000' ,pct:'single.jpg'}) return n
```

Pada proses penambahan tersebut *neo4j* dapat mengeksekusi data pada 231ms sedangkan *mysql* dapat mengeksekusi data pada 0.089sec.

b. Proses Update data

1) Pada *MySql:*

```
UPDATE `hotel`.`kamar` SET `NAMA_KAMAR`='Deluxe room' WHERE `ID_KAMAR`='1';
```

2) Pada *Neo4j*

```
match (n:kamar) where n.nomor='K1' set n.namakmr='Interior Room' return n
```

Pada proses update data tersebut menunjukan bahwa kecepatan yang dimiliki *mysql* adalah 0.093*sec*. sedangkan untuk *neo4j* adalah 869*ms*.

c. Proses Hapus

1) Pada *MmySql*

```
DELETE FROM `hotel`.`kamar` WHERE `ID_KAMAR`='1';
```

2) Pada *Neo4j*

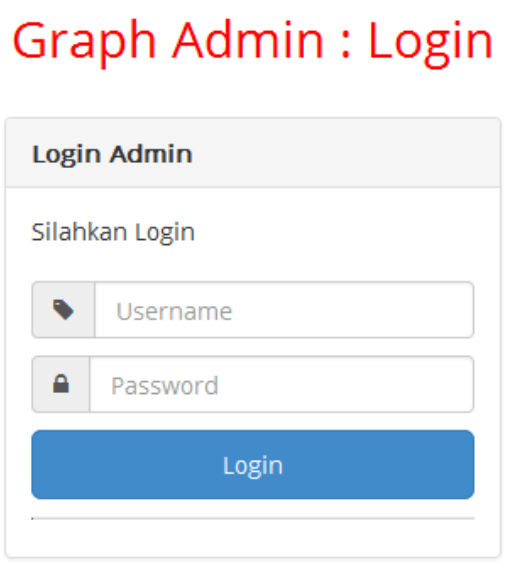
```
Match (n:kamar) where n.nomor='K1' detach delete n
```

Pada Mysql menghapus satu baris yang ingin dihapus. Sedangkan pada Neo4j menhapus satu *node* dan juga menghapus relasi pada *node* tersebut. Untuk waktu eksekusi pada *Mysql* adalah 0.124*sec*. sedangkan pada *Neo4*j adalah 372*ms*.

HASIL DAN PEMBAHASAN

1. Halaman Login

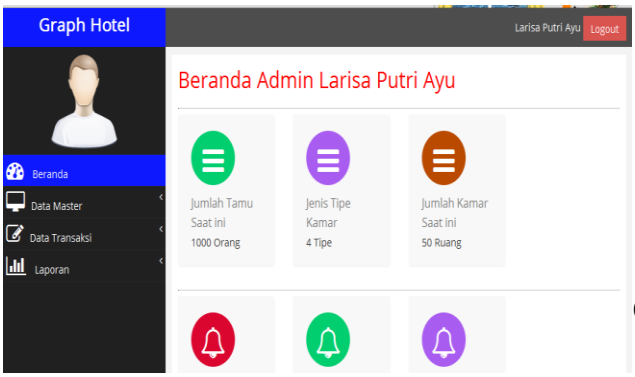
Pada halaman ini admin harus memasukkan *username* dan *password* untuk dapat masuk pada aplikasi. Apabila *username* dan *password* salah, maka akan muncul peringatan. Untuk *login* terdapat berbagai menu yang dapat diakses oleh *user* dengan hak akses yang berbeda dan sesuai dengan masing-masing level *user*.



Gambar 8. Tampilan Halaman Login

2. Halaman Beranda

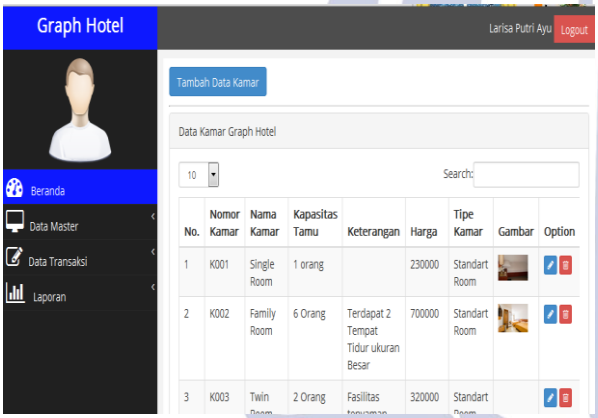
Pada Sistem Informasi Hotel ni, tampilan utama yang keluar adalah halaman utama admin. Pada halaman utama ini terdapat beberapa menu yaitu master data, transaksi dan laporan.



Gambar 9. Halaman Beranda

3. Halaman Master Kamar

Halaman ini hanya dimiliki oleh admin. Halaman ini terdapat tambah, edit, dan hapus untuk pengaturan pengguna sistem ini.

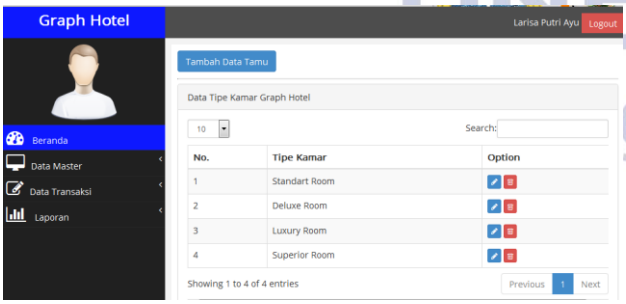


Gambar 10. Tampilan Halaman Master Kamar

4. Halaman Master Tipe Kamar

Halaman ini hanya dimiliki oleh admin. Halaman ini terdapat tambah, edit, dan hapus untuk pengaturan pengguna sistem ini.

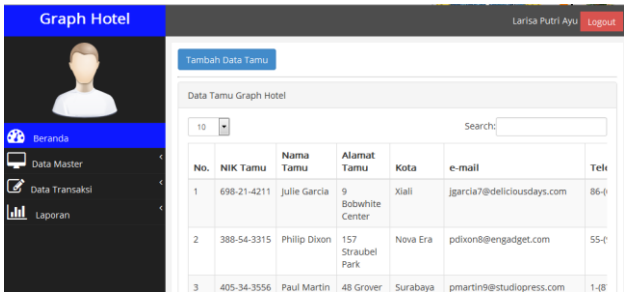
.



Gambar 11. Tampilan Halaman Master Tipe Kamar

5. Halaman Master Tamu

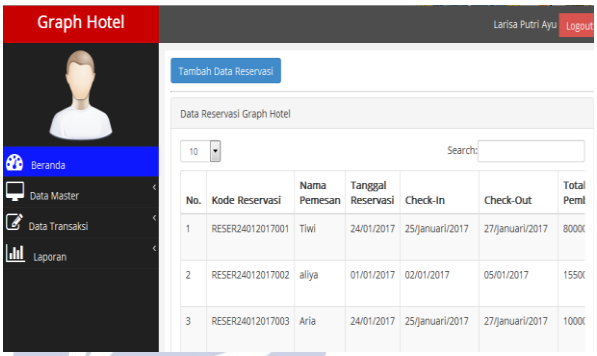
Halaman ini hanya dimiliki oleh admin. Halaman ini terdapat tambah, edit, dan hapus untuk pengaturan pengguna sistem ini. beberapa fitur untuk menambahkan data tamu. Selain itu pengguna juga dapat mengubah dan menghapus data.



Gambar 12. Tampilan Halaman Master Tamu

6. Proses Data Transaksi *Reservasi*

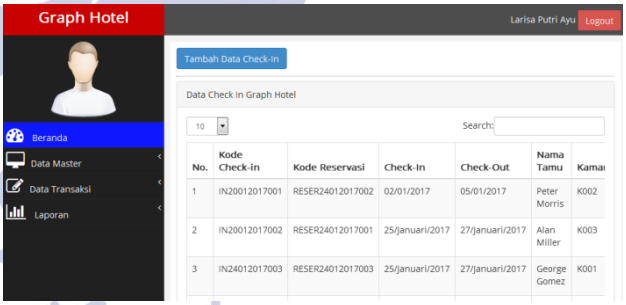
Pada halaman transaksi reservasi tamu terhadap hotel. Halaman ini bisa diakses oleh admin.



Gambar 13 Tampilan Halaman Proses Transaksi *Reservasi*

7. Proses Transaksi *Check-In* Tamu

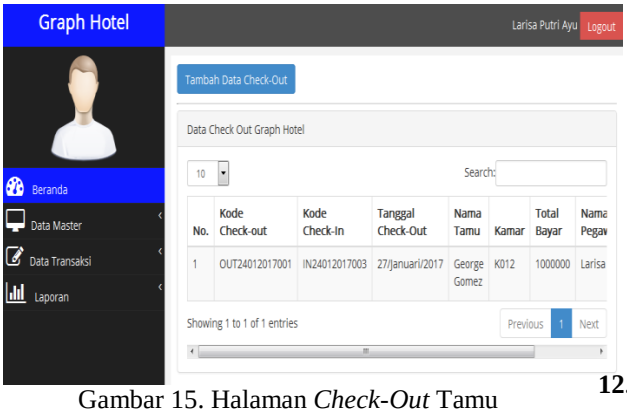
Pada halaman ini berisi tentang data transaksi check in yang pernah dilakukan oleh tamu pada hotel. Halaman ini bisa diakses oleh admin



Gambar 14. Tampilan Halaman Proses Transaksi *Check-In* Tamu

8. Proses Transaksi *Check-Out* Tamu

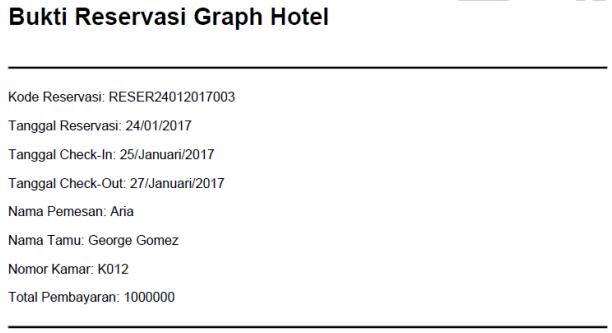
Pada halaman transaksi check-out tamu terhadap hotel. Halaman ini bisa diakses oleh admin.



Gambar 15. Halaman *Check-Out* Tamu

9. Bukti Transaksi *Resevasi*

Pada halaman ini merupakan hasil cetak bukti *reservasi* yang dilakukan oleh admin unutuk tamu.



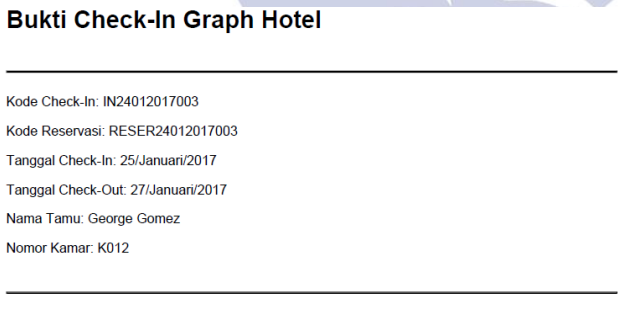
Bukti Reservasi Graph Hotel

Kode Reservasi: RESER24012017003
Tanggal Reservasi: 24/01/2017
Tanggal Check-In: 25/Januari/2017
Tanggal Check-Out: 27/Januari/2017
Nama Pemesan: Aria
Nama Tamu: George Gomez
Nomor Kamar: K012
Total Pembayaran: 1000000

Gambar 16. Bukti cetak data *Reservasi*

10. Bukti Transaksi *Check-In*

Pada halaman ini merupakan hasil cetak bukti *check-in* yang dilakukan oleh admin unutuk tamu.



Bukti Check-In Graph Hotel

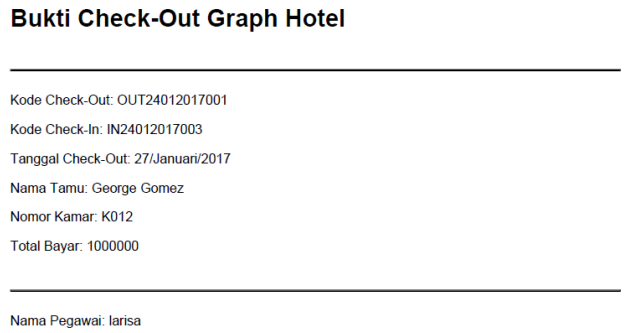
Kode Check-In: IN24012017003
Kode Reservasi: RESER24012017003
Tanggal Check-In: 25/Januari/2017
Tanggal Check-Out: 27/Januari/2017
Nama Tamu: George Gomez
Nomor Kamar: K012

Nama Pegawai: larisa

Gambar 17. Bukti cetak data *check-in*

11. Bukti Transaksi *Check-Out*

Pada halaman ini merupakan hasil cetak bukti *check-out* yang dilakukan oleh admin unutuk tamu.



Bukti Check-Out Graph Hotel

Kode Check-Out: OUT24012017001
Kode Check-In: IN24012017003
Tanggal Check-Out: 27/Januari/2017
Nama Tamu: George Gomez
Nomor Kamar: K012
Total Bayar: 1000000

Nama Pegawai: larisa

Gambar 18. Bukti cetak data *check-ot*

12. Hasil perbandingan *database*

Tabel 1 merupakan tabel perbadingan waktu antara *database Neo4j* dengan *database MySql*. Dari hasil rata – rata tersebut membuktikan bahwa *database MySql* lebih cepat dan stabil dibandingkan *database Neo4j*.

Tabel 1. Hasil perbandingan *database*

Membandingkan	Perbandingan waktu	
	Neo4j	*MySql*
Menambahkan data	231ms	89ms
Mengedit data	869*ms*	93*ms*
Menghapus data	372*ms*	124*ms*
Merelasikan data	942*ms*	890*ms*
Menampilkan 100 data *Check-in*	387*ms*	432*ms*
Menampilkan 100 data *check-out*	129*ms*	109*ms*
Rata - rata	488,33*ms*	289,5*ms*

PENUTUP

Simpulan

Kesimpulan dari Sistem Informasi Hotel dapat membangun susatu sistem informasi mengunakan *database Neo4j*. Akan tetapi perbandingan antara waktu pemeroses antara *databaase Neo4j* dengan *database MySql* memiliki kecepatan sendiri. Kecepatan rata –rata yang dimiliki *database MySql* adalah 289,5*ms*. Sedangkan *database Neo4j* memiliki kecepatan 488,33ms. *Kecepatan* tersebut membuktikan *database MySql* lebih cepat dibandingkan dengan *database Neo4j*. Keuntungan mengunakan *database Neo4j* adalah relasi antar *node* dapat dibuat tanpa membutuhkan banyak operasi *JOIN TABLE* seperti pada *MySQL*. Sedangkan kelemahannya,database *Neo4J* masih baru sehingga perlu ada pengembangan lebih jauh untuk dapat menggunakannya.

Saran

1. Proses *reservasi* melalui tamu langsung dapat dilakukan oleh tamu tanpa harus pergi ke hotel.
2. Penambahan proses tambah kamar untuk tamu supaya dapat melakukan *reservasi* kamar lebih dari satu.
3. Penambahan fitur hitung pemakaian laundry, restoran, dan lain-lain.

4. Perubahan tampilan agar telihat lebih mudah untuk digunakan.

DAFAR PUSTAKA

Abidin Ali, dkk. (2012). Pencarian Dengan Knowledge Graph. *Prosiding Seminar Ilmiah Nasional Komputer dan Sistem Intelijen (KOMMIT 2012)* (pp. 11-17). Depok: Universitas Gunadarma.

Kolomicenko dan Vojtech. (2013). *Analysis and Experimental Comparison of Graph Databases.* Prague: Charles University.

Neo4J Team. (2016, Oktober 05). Retrieved from https://neo4j.com/blog/rdbms-neo4j-etl-tool/

Yubin Park, Mallikarjun Shankar,dan Joydeep Ghosh. (2014). *Graph Database in Large Scale Healthcare System. A Proposal for Efficient Data Management and Utilization.* Austin: University of Texas.